

《数据分析》课外实践报告

项 目 名 称： **学生信息管理平台**

所 在 班 级：  **2022级计算机科学与技术三班**

第**5**组 成员：张田兴 侯栩涛 陈家祝 刘祥培 薛琦嘉 吴昕炜

彭文强 黄征

指 导 教 师：  **王鸣涛**

成 绩 评 定：

起 止 时 间： **2024年11月26日——2024年12月4日**

项目基本信息

|  |  |
| --- | --- |
| 项目名称 |  |
| 项目简介 | **1 项目描述：**  实现一个学生信息管理平台，该平台需具备以下功能：添加学生信息、移除学生信息、更新学生信息以及检索学生信息。   1. **实现的功能及结果：**   **2.1 主要功能：**  （1）学生信息的插入  （2）学生信息的删除  （3）学生信息的修改  （4）学生信息的查询  **2.2 结果：**  实现了添加学生信息、移除学生信息、更新学生信息以及检索学生信息等功能。   1. **优缺点：**   **3.1优点：**  构建一个简洁明了、逻辑直观的架构体系，使得其基础设计易于理解且直观清晰。   * 1. **缺点：**   通过编写代码来逐一创建界面上的各个元素，并妥善安排它们的布局是一个相对复杂且细致的任务。 |

1. 系统描述

学生信息管理系统是一个采用Python语言开发的应用程序，其主要目标是协助教师或学校管理人员更加高效地管理学生数据。为了提供一个直观且易用的操作界面，该系统借助了Tkinter库来打造一个用户友好的图形化界面。通过这个界面，用户可以轻松地通过点击和输入来完成添加、编辑、删除以及查询学生信息的操作。此外，为了进一步提升用户的操作效率，该系统还集成了keyboard库（如果已应用），从而实现了快捷键操作的功能。

1. 系统设计

本系统致力于打造一个基于Python语言的学生信息管理解决方案，该方案通过构建图形用户界面（GUI）来提供一系列学生信息管理功能，包括添加、编辑、删除和查询等。我们的设计目标是确保系统具备出色的用户交互体验与数据安全保障，以满足学校或培训机构在日常管理学生信息方面的需求。

在系统架构上，我们采用了客户端/服务器（C/S）的设计模式，但鉴于本系统定位为单机版应用，服务器部分被优化为本地数据库管理系统。系统由以下几个核心模块构成：

1、用户界面模块：我们利用Tkinter库精心打造了一个直观易用的图形用户界面。该界面包括一个功能齐全的主界面，用于触发各种操作；一个信息输入编辑界面，供用户输入或修改学生信息；以及一个查询结果界面，用于展示检索到的学生信息。

2、业务逻辑模块：此模块负责处理用户通过界面输入的数据，并根据用户需求执行相应的添加、编辑、删除和查询操作。同时，它还与数据库模块进行交互，确保数据的准确性与一致性。

3、数据库模块：我们选择了mysql等轻量级数据库来存储学生信息，以确保数据的安全性和持久性。通过数据库管理，我们能够高效地管理大量学生数据，并保障数据的完整性和可靠性。

4、快捷键处理模块（可选）：为了提升用户体验，我们还集成了keyboard库来监听键盘事件，并实现了快捷键操作功能。这样，用户可以通过简单的键盘组合来快速触发常用操作，提高工作效率。

在主界面设计上，我们精心布置了“添加学生”、“编辑学生”、“删除学生”和“查询学生”等功能按钮，以及一个用于显示当前系统状态或提示信息的区域。信息输入/编辑界面则提供了学号、姓名、性别、年龄和班级等关键字段的输入框，以及“添加”、“保存”和“取消”等按钮，方便用户进行信息的录入和修改。查询结果界面则清晰展示了检索到的学生信息列表，包括学号、姓名和班级等基本信息，并提供了查看详细信息的按钮，以满足用户的多样化需求。

1. 系统实现

开发Python学生信息管理系统包括三部分：实现功能、进行测试与优化。

1、系统开发阶段

我们将使用Python语言，结合Tkinter库来构建图形用户界面，实现一个学生信息管理系统的基本功能，包括添加、编辑、删除和查询学生信息。同时，我们会利用mysql数据库来存储和管理这些数据，确保数据的持久性和安全性。此外，如果条件允许，我们还将集成keyboard库来支持快捷键操作，提高用户体验。

在这一阶段，我们将编写代码来实现以下模块：

用户界面模块：使用Tkinter构建直观易用的图形界面，包括主界面、信息输入/编辑界面和查询结果界面。

业务逻辑模块：处理用户输入的数据，执行相应的数据库操作，并返回结果给用户。

数据库模块：使用mysql数据库存储学生信息，提供数据访问和操作的接口。

快捷键处理模块（可选）：使用keyboard库监听键盘事件，实现快捷键功能。

2、测试阶段

在系统开发完成后，我们将编写测试用例来验证系统的功能和稳定性。这些测试用例将涵盖系统的所有功能点，包括添加、编辑、删除和查询学生信息等。我们将通过模拟用户操作来触发这些功能，并检查系统的响应是否符合预期。同时，我们还会对系统的性能进行测试，以确保其能够满足实际应用场景的需求。

在测试过程中，我们可能会发现一些潜在的问题或缺陷。针对这些问题，我们将及时记录并进行分析，然后制定相应的修复方案。

3、优化与改进阶段

根据测试结果，我们将对系统进行优化和改进。这包括修复发现的问题、优化代码性能、提高用户体验等方面。例如，我们可以对数据库查询进行优化，以减少响应时间；对界面布局进行调整，使其更加美观和易用；或者增加更多的快捷键操作，以提高用户的工作效率。

在这一阶段，我们将持续关注系统的运行状态，并根据用户反馈和需求进行迭代更新。通过不断优化和改进，我们将努力打造一个功能强大、稳定可靠且用户友好的学生信息管理系统。

1. 系统演示及使用说明

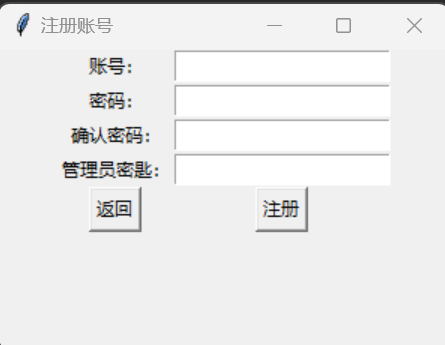
**系统演示：**

1、运行后首先进入登录界面：



系统默认有一套账号，假如初始不进行账号预输入，就会自动加入一套账号为：root，密码为：123456的账号，使用该账号即可登录。

2、点击注册后进入注册页面：



管理员密匙为root，按照格式输入账号成功注册后即可进入到主页面。

3、添加功能



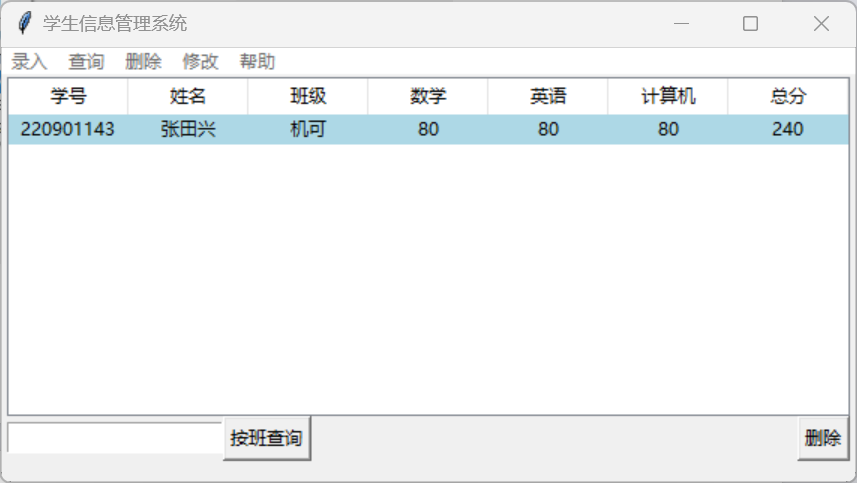
初始加载页面为录入页面，输入对应的信息后点击录入，假如格式正确即可成功录入信息，假如格式不正确则会报错，点击清空即可将输入的信息全部清除。

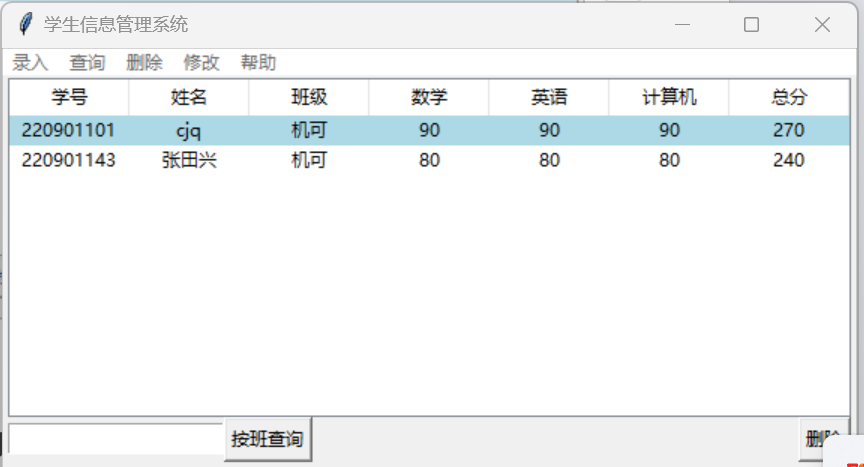
4、查询功能

查询页面如下：



会弹出一个下拉列表，显示查询页面中信息的排序方法。





5、删除功能

进入删除界面后：

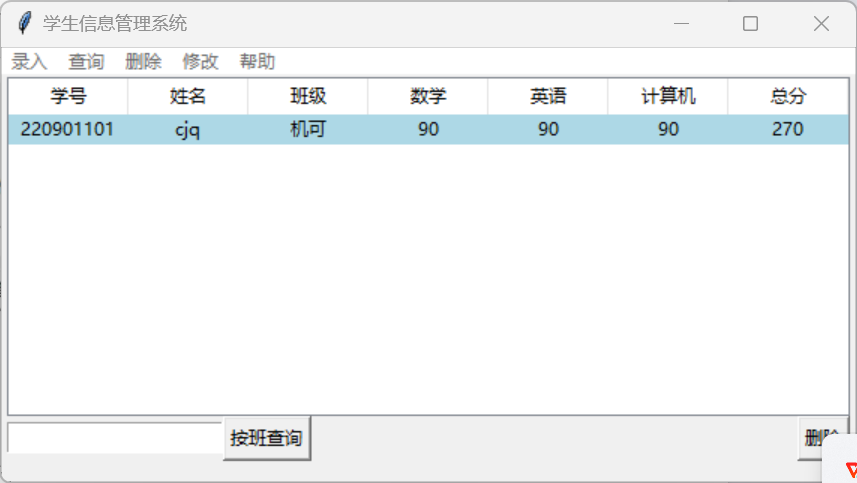
可选择查询方法，通过学号或姓名来查询，查询到就会显示对应的各个信息，查询不到就会提示：





点击删除后，信息便会删除，可以去查询页面查看：





删除成功。

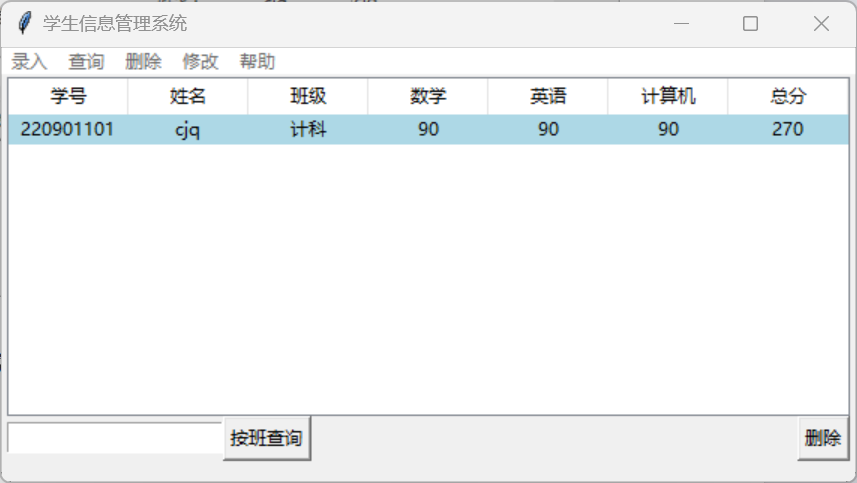
6、修改功能

进入修改页面：



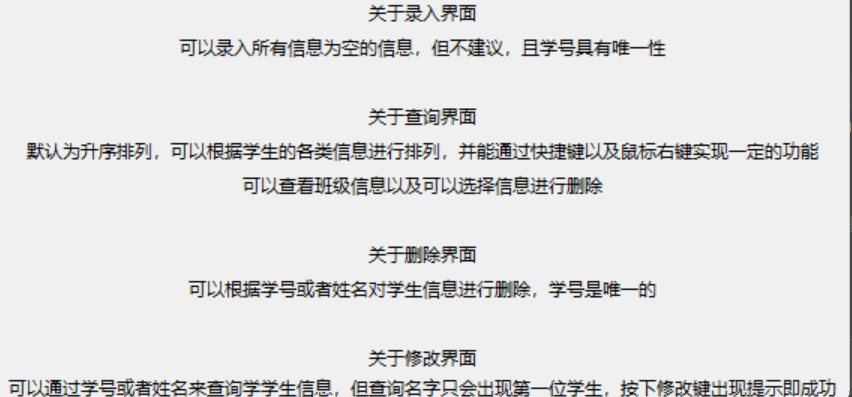
查询到后在输入框输入想要修改为的值即可成功修改：





**系统使用说明：**

最后的帮助页面是该系统的一些使用说明



1. 问题及解决方法

问题一：界面布局难题

在初始阶段，我们采用了tkinter库的grid布局方式来设计用户界面，然而，实际操作中，我们遇到了界面布局难以达到预期效果的挑战。

解决方案：

为了克服这一难题，我们决定转而使用tkinter库的place布局方式，以期获得更加灵活和理想的界面布局效果。

问题二：数据库连接障碍

当我们着手进行后端开发，并尝试与SQL Server数据库建立连接时，却遭遇了因SQL Server采用Windows主机验证机制而导致的连接障碍，这无疑给我们的开发工作带来了不小的困扰。

解决方案：

为了绕过这一障碍，我们决定改用mysql数据库。mysql数据库以其广泛的兼容性和易用性，为我们提供了更为顺畅的数据库连接和开发体验。

1. 课外实践总结

本次课外实践中，我们小组齐心协力，利用Python编程语言和MySQL数据库技术，共同打造了一个功能完备的学生信息管理系统。在这个过程中，我们不仅深入学习了Python的编程技巧，还熟练掌握了MySQL数据库的操作与管理。更重要的是，这次实践极大地锻炼了我们的团队合作能力，让我们学会了如何在团队中分工合作、共同解决问题。通过不断的交流与协作，我们成功地将理论知识转化为实际成果，收获颇丰。这次实践不仅提升了我们的专业技能，更为我们未来的学习和工作积累了宝贵的经验。

1. 源代码

**此项目包含四个文件：**

**一、mainPage:**

from tkinter import \*

from views import ChangeFrame,DeleteFrame,InsertFrame,SearchFrame,HelpFrame

import sql

import keyboard

class MainPage:

def \_\_init\_\_(self,master):

self.root = master

self.root.title('学生信息管理系统')

self.root.geometry('570x290')

self.create\_page()

def create\_page(self):

self.insert\_frame = InsertFrame(self.root)

self.search\_frame = SearchFrame(self.root)

self.delete\_frame = DeleteFrame(self.root)

self.change\_frame = ChangeFrame(self.root)

self.help\_frame = HelpFrame(self.root)

menubar = Menu(self.root,tearoff=False)

menubar.add\_command(label='录入',command=self.show\_insert)

submenu\_search = Menu(menubar)

submenu\_search.add\_command(label='降序',command=self.show\_ssd,accelerator="Ctrl + J")

submenu\_search.add\_separator()

submenu\_search.add\_command(label='学号',command=self.show\_search\_id,accelerator="Ctrl + J")

submenu\_search.add\_command(label='总分',command=self.show\_search\_total,accelerator="Ctrl + T")

submenu\_search.add\_command(label='数学',command=self.show\_search\_math,accelerator="Ctrl + M")

submenu\_search.add\_command(label='英语',command=self.show\_search\_english,accelerator="Ctrl + E")

submenu\_search.add\_command(label='计算机',command=self.show\_search\_computer,accelerator="Ctrl + S")

keyboard.add\_hotkey('ctrl+j',self.show\_ssd)

keyboard.add\_hotkey('ctrl+d',self.show\_search\_id)

keyboard.add\_hotkey('ctrl+t',self.show\_search\_total)

keyboard.add\_hotkey('ctrl+m',self.show\_search\_math)

keyboard.add\_hotkey('ctrl+e',self.show\_search\_english)

keyboard.add\_hotkey('ctrl+s',self.show\_search\_computer)

menubar.add\_cascade(label='查询',menu=submenu\_search)

menubar.add\_command(label='删除',command=self.show\_delete)

menubar.add\_command(label='修改',command=self.show\_change)

menubar.add\_command(label='帮助',command=self.show\_help)

self.root.config(menu = menubar)

def xShowMenu(event):

menubar.post(event.x\_root,event.y\_root)

self.root.bind("<Button-3>",xShowMenu)

self.show\_insert()

def show\_ssd(self):

sql.sort\_stu ^=1

self.show\_search()

def show\_search\_id(self):

sql.sort\_data = 0

self.show\_search()

def show\_search\_total(self):

sql.sort\_data = 1

self.show\_search()

def show\_search\_math(self):

sql.sort\_data = 2

self.show\_search()

def show\_search\_english(self):

sql.sort\_data = 3

self.show\_search()

def show\_search\_computer(self):

sql.sort\_data = 4

self.show\_search()

def show\_insert(self):

self.insert\_frame.pack()

self.search\_frame.pack\_forget()

self.delete\_frame.pack\_forget()

self.change\_frame.pack\_forget()

self.help\_frame.pack\_forget()

def show\_search(self):

self.insert\_frame.pack\_forget()

self.delete\_frame.pack\_forget()

self.change\_frame.pack\_forget()

self.help\_frame.pack\_forget()

self.search\_frame.pack()

self.search\_frame.show\_search()

def show\_delete(self):

self.delete\_frame.pack()

self.insert\_frame.pack\_forget()

self.search\_frame.pack\_forget()

self.change\_frame.pack\_forget()

self.help\_frame.pack\_forget()

def show\_change(self):

self.change\_frame.pack()

self.insert\_frame.pack\_forget()

self.search\_frame.pack\_forget()

self.delete\_frame.pack\_forget()

self.help\_frame.pack\_forget()

def show\_help(self):

self.change\_frame.pack\_forget()

self.insert\_frame.pack\_forget()

self.search\_frame.pack\_forget()

self.delete\_frame.pack\_forget()

self.help\_frame.pack()

**二、LoginPage:**

from tkinter import \*

from tkinter import messagebox

import sql

from mainPage import MainPage

class LoginPage:

def \_\_init\_\_(self,master):

self.root = master

self.root.title('登录页面')

self.root.geometry('300x150')

self.page = Frame(self.root)

self.page.pack()

self.username = StringVar()

self.password = StringVar()

# 设计窗体内容：

Label(self.page,text='账号：').grid(row=1,column=1,pady=5)

Entry(self.page,textvariable=self.username).grid(row=1,column=2)

Label(self.page, text='密码：').grid(row=2, column=1, pady=5)

Entry(self.page, show='\*',textvariable=self.password).grid(row=2, column=2)

Button(self.page,text='注册',command=self.register).grid(row=3,column=1,pady=4)

Button(self.page,text='登录',command=self.login).grid(row=3,column=2,pady=4)

Button(self.page,text='退出',command=self.page.quit).grid(row=3,column=3,pady=4)

def register(self):

self.page.destroy()

add\_login(self.root)

def login(self):

uname = self.username.get()

pwd = self.password.get()

flag, message = sql.check\_login(uname,pwd)

if flag:

self.page.pack\_forget()

MainPage(self.root)

else: messagebox.showwarning(title= '警告',message = message)

class add\_login:

def \_\_init\_\_(self,master):

self.page = master

self.username = StringVar()

self.passowrd = StringVar()

self.passowrd\_queren = StringVar()

self.passowrd\_root = StringVar()

self.page.title('注册账号')

self.page.geometry('300x200')

self.root = Frame(self.page)

self.root.pack()

Label(self.root,text='账号：').grid(row = 1,column= 1)

Entry(self.root,textvariable=self.username).grid(row=1,column=2)

Label(self.root, text='密码：').grid(row=2, column=1)

Entry(self.root,show='\*',textvariable=self.passowrd).grid(row=2, column=2)

Label(self.root,text='确认密码：').grid(row=3, column=1)

Entry(self.root,show='\*',textvariable=self.passowrd\_queren).grid(row=3, column=2)

Label(self.root, text='管理员密匙：').grid(row=4, column=1)

Entry(self.root,show='\*',textvariable=self.passowrd\_root).grid(row=4, column=2)

Button(self.root,text='返回',command=self.login\_page).grid(row=5,column=1)

Button(self.root,text='注册',command=self.examine).grid(row=5,column=2)

def login\_page(self):

self.root.pack\_forget()

LoginPage(self.page)

def examine(self):

self.uname = self.username.get()

self.pwd = self.passowrd.get()

self.pwd\_qr = self.passowrd\_queren.get()

self.pwd\_r = self.passowrd\_root.get()

if len(self.uname) < 5:

messagebox.showwarning(title='警告',message='账号不符合要求,请输入至少五位字符')

elif sql.check\_username(self.uname) == True:

messagebox.showwarning(title='警告',message='该账号名已存在')

elif len(self.pwd) < 8:

messagebox.showwarning(title='警告',message='密码不符合要求，请输入至少八位密码')

elif self.pwd != self.pwd\_qr:

messagebox.showwarning(title='警告',message='请保持密码与确认密码一致')

elif self.pwd\_r !='root':

messagebox.showwarning(title='警告',message='管理员密匙错误')

else:self.login()

def login(self):

sql.add\_admin(self.uname,self.pwd)

self.root.pack\_forget()

MainPage(self.page)

if \_\_name\_\_ == '\_\_main\_\_':

page = Tk()

LoginPage(page)

page.mainloop()

**三、Views:**

from tkinter import \*

from tkinter import ttk

import sql

from tkinter import messagebox

class InsertFrame(Frame):

def \_\_init\_\_(self,root):

super().\_\_init\_\_(root)

self.id = StringVar()

self.name = StringVar()

self.class1 = StringVar()

self.math = StringVar()

self.english = StringVar()

self.computer = StringVar()

self.status\_insert = StringVar()

self.insert\_page()

def insert\_page(self):

Label(self,text='学号：').grid(row=1,column=1,pady=5)

self.entry\_id = Entry(self,textvariable=self.id)

self.entry\_id.grid(row=1,column=2,pady=5)

Label(self, text='姓名：').grid(row=2, column=1, pady=5)

self.entry\_name = Entry(self, textvariable=self.name)

self.entry\_name.grid(row=2, column=2, pady=5)

Label(self, text='班级：').grid(row=3, column=1, pady=5)

self.entry\_class = Entry(self, textvariable=self.class1)

self.entry\_class.grid(row=3, column=2, pady=5)

Label(self, text='数学：').grid(row=4, column=1, pady=5)

self.entry\_math = Entry(self, textvariable=self.math)

self.entry\_math.grid(row=4, column=2, pady=5)

Label(self, text='英语：').grid(row=5, column=1, pady=5)

self.entry\_english = Entry(self, textvariable=self.english)

self.entry\_english.grid(row=5, column=2, pady=5)

Label(self, text='计算机：').grid(row=6, column=1, pady=5)

self.entry\_com = Entry(self, textvariable=self.computer)

self.entry\_com.grid(row=6, column=2, pady=5)

Button(self,text='清空',command=self.insert\_clear).grid(row=7,column=1,pady=10)

Button(self,text='录入',command=self.insert\_data).grid(row=7,column=3,pady=10)

Label(self,textvariable=self.status\_insert).grid(row=8,column=2,padx=10)

def insert\_data(self):

if not self.id.get():

self.insert\_id=int(0)

else:

self.insert\_id=int(self.id.get())

if not self.name.get():

self.insert\_name='NULL'

else:

self.insert\_name=self.name.get()

if not self.class1.get():

self.insert\_class='NULL'

else:

self.insert\_class=self.class1.get()

if not self.math.get():

self.insert\_math=int(0)

else:

self.insert\_math=int(self.math.get())

if not self.english.get():

self.insert\_english=int(0)

else:

self.insert\_english=int(self.english.get())

if not self.computer.get():

self.insert\_computer=int(0)

else:

self.insert\_computer=int(self.computer.get())

flag,s = sql.check\_id(self.insert\_id)

self.status\_insert.set(s)

if flag == False:

return

self.insert\_total = self.insert\_math + self.insert\_computer + self.insert\_english

stu = (self.insert\_id,self.insert\_name,self.insert\_class,self.insert\_math,self.insert\_english

,self.insert\_computer,self.insert\_total)

sql.insert(stu)

def insert\_clear(self):

self.entry\_id.delete(0,END)

self.entry\_name.delete(0,END)

self.entry\_class.delete(0,END)

self.entry\_math.delete(0,END)

self.entry\_english.delete(0,END)

self.entry\_com.delete(0,END)

class SearchFrame(Frame):

def \_\_init\_\_(self,root):

super().\_\_init\_\_(root)

self.table\_search=Frame()

self.show\_table()

def show\_table(self):

columns = ("id","name","class1","math","english","computer","total")

columns\_values = ("学号","姓名","班级","数学","英语","计算机","总分")

self.tree\_view = ttk.Treeview(self,show='headings',columns = columns)

for col in columns:

self.tree\_view.column(col,width=80,anchor='center')

for col,colvalue in zip(columns,columns\_values):

self.tree\_view.heading(col,text=colvalue)

self.tree\_view.pack(fill = BOTH,expand = True)

self.show\_search()

self.class\_class = StringVar()

Entry(self,textvariable=self.class\_class).pack(side = LEFT)

Button(self,text='按班查询',command=self.search\_class).pack(side=LEFT)

Button(self,text='删除',command=self.treeviewClick).pack(side = RIGHT)

def treeview\_sort\_column1(tv,col,reverse):

l = [(tv.set(k,col),k) for k in tv.get\_children('')]

l.sort(key=lambda t:int(t[0]),reverse=reverse)

for index,(val,k) in enumerate(l):

tv.move(k,'',index)

tv.heading(col, command=lambda:treeview\_sort\_column1(tv,col,not reverse))

self.tree\_color()

def treeview\_sort\_column2(tv,col,reverse):

l = [(tv.set(k,col),k) for k in tv.get\_children('')]

l.sort(reverse=reverse)

for index,(val,k) in enumerate(l):

tv.move(k,'',index)

tv.heading(col,command=lambda: treeview\_sort\_column2(tv,col,not reverse))

self.tree\_color()

for i in range(7):

if i >=1 and i<=2:

self.tree\_view.heading(columns[i],text=columns\_values[i],

command=lambda \_col=columns[i]: treeview\_sort\_column2(self.tree\_view,\_col,

False))

else:

self.tree\_view.heading(columns[i], text=columns\_values[i],

command=lambda \_col=columns[i]: treeview\_sort\_column2(self.tree\_view, \_col,

False))

self.tree\_view.tag\_configure('even',background='lightblue')

def search\_class(self):

for \_ in map(self.tree\_view.delete, self.tree\_view.get\_children('')):

pass

if not self.class\_class.get():

self.show\_search()

return

else:

self.class\_value = self.class\_class.get()

students = sql.search\_class(self.class\_value)

index = -1

for stu in students:

self.tree\_view.insert('',index + 1,values=

(stu['id'],stu['name'],stu['class'],stu['math'],

stu['english'],stu['computer'],stu['total']

))

self.tree\_color()

def treeviewClick(self):

for item in self.tree\_view.selection():

item\_text = self.tree\_view.item(item,'values')

sql.delete\_id(item\_text[0])

self.show\_search()

def show\_search(self):

for \_ in map(self.tree\_view.delete,self.tree\_view.get\_children('')):

pass

students = sql.all()

index = -1

for stu in students:

self.tree\_view.insert('',index + 1,values=(

stu['id'],stu['name'],stu['class'],stu['math'],

stu['english'],stu['computer'],stu['total']

))

self.tree\_color()

def tree\_color(self):

items = self.tree\_view.get\_children()

i = 0

for hiid in items:

if i/2 != int(i/2):

tag1=''

else:

tag1 = 'even'

self.tree\_view.item(hiid,tag=tag1)

i += 1

class DeleteFrame(Frame):

def \_\_init\_\_(self,root):

super().\_\_init\_\_(root,width = 570,height = 290)

self.delete\_student = StringVar()

self.status\_student = StringVar()

Label(self,text='请输入需要删除学生的').place(x = 40,y = 60)

Label(self,text='姓名或者学号').place(x = 64,y = 80)

Entry(self,textvariable=self.delete\_student).place(x = 30,y = 100)

Button(self,text='按学号查询',command=self.id\_delete).place(x = 30,y = 130)

Button(self,text='按姓名查询',command=self.name\_delete).place(x = 110,y = 130)

Label(self,textvariable=self.status\_student).place(x = 45,y = 160)

self.id = StringVar()

self.name = StringVar()

self.class1 = StringVar()

self.math = StringVar()

self.english = StringVar()

self.computer = StringVar()

Label(self,text='学号：').place(x=300,y=20)

Label(self,textvariable=self.id).place(x=360,y=20)

Label(self,text='姓名：').place(x=300,y=50)

Label(self,textvariable=self.name).place(x=360,y=50)

Label(self,text='班级：').place(x=300,y=80)

Label(self,textvariable=self.class1).place(x=360,y=80)

Label(self,text='数学：').place(x=300,y=110)

Label(self,textvariable=self.math).place(x=360,y=110)

Label(self,text='英语：').place(x=300,y=140)

Label(self,textvariable=self.english).place(x=360,y=140)

Label(self,text='计算机：').place(x=300,y=170)

Label(self,textvariable=self.computer).place(x=360,y=170)

self.status\_delete = StringVar()

Button(self,text='删除',command=self.delete\_stu).place(x=340,y=210)

Label(self,textvariable=self.status\_delete).place(x=300,y=250)

def id\_delete(self):

if self.delete\_student.get():

self.search\_user\_id = self.delete\_student.get()

flag,stu = sql.search\_id(self.search\_user\_id)

if flag:

self.id.set(stu[0][0]),self.name.set(stu[0][1])

self.class1.set(stu[0][2]),self.math.set(stu[0][3])

self.english.set(stu[0][4]),self.computer.set(stu[0][5])

self.status\_student.set('数据查询成功')

else:

self.status\_student.set(stu)

else:

self.status\_student.set('请输入需要查询的信息')

def name\_delete(self):

if self.delete\_student.get():

self.search\_user\_name = self.delete\_student.get()

flag,stu = sql.search\_name(self.search\_user\_name)

if flag:

self.id.set(stu[0][0])

self.name.set(stu[0][1])

self.class1.set(stu[0][2])

self.math.set(stu[0][3])

self.english.set(stu[0][4])

self.computer.set(stu[0][5])

self.status\_student.set('数据查询成功')

else:

self.status\_student.set(stu)

else:

self.status\_student.set('请输入需要查询的信息')

def delete\_stu(self):

flag,str = sql.delete\_id(self.id.get())

if not self.id.get():

str = '需要删除信息不能为NULL'

self.status\_delete.set(str)

class ChangeFrame(Frame):

def \_\_init\_\_(self,root):

super().\_\_init\_\_(root,width=570,height = 290)

self.change\_student = StringVar()

self.status\_student = StringVar()

self.status\_name = StringVar()

self.id = StringVar()

self.name = StringVar()

self.class1 = StringVar()

self.math = StringVar()

self.english = StringVar()

self.computer = StringVar()

self.id\_change\_before = StringVar()

self.name\_change\_before = StringVar()

self.class1\_change\_before = StringVar()

self.math\_change\_before = StringVar()

self.english\_change\_before = StringVar()

self.computer\_change\_before = StringVar()

self.insert\_page()

def insert\_page(self):

Label(self,text='请输入需要查询学生的').place(x=40,y=60)

Label(self,text='姓名或学号').place(x=64,y=80)

Entry(self,textvariable=self.change\_student).place(x=30,y=100)

Button(self,text='按学号查询',command=self.id\_change).place(x=30,y=130)

Button(self,text='按姓名查询',command=self.name\_change).place(x=110,y=130)

Label(self,textvariable=self.status\_student).place(x=45,y=160)

Label(self,text='学号：').place(x=240,y=20)

Label(self,textvariable=self.id\_change\_before).place(x=320,y=20)

self.entry\_id = Entry(self,textvariable=self.id)

self.entry\_id.place(x=380,y=20)

Label(self,text='姓名：').place(x=240,y=50)

Label(self,textvariable=self.name\_change\_before).place(x=315,y=50)

self.entry\_name = Entry(self,textvariable=self.name)

self.entry\_name.place(x=380,y=50)

Label(self, text='班级：').place(x=240, y=80)

Label(self, textvariable=self.class1\_change\_before).place(x=315, y=80)

self.entry\_class = Entry(self, textvariable=self.class1)

self.entry\_class.place(x=380, y=80)

Label(self, text='数学：').place(x=240, y=110)

Label(self, textvariable=self.math\_change\_before).place(x=315, y=110)

self.entry\_math = Entry(self, textvariable=self.math)

self.entry\_math.place(x=380, y=110)

Label(self, text='英语：').place(x=240, y=140)

Label(self, textvariable=self.english\_change\_before).place(x=315, y=140)

self.entry\_english = Entry(self, textvariable=self.english)

self.entry\_english.place(x=380, y=140)

Label(self, text='计算机：').place(x=240, y=170)

Label(self, textvariable=self.computer\_change\_before).place(x=315, y=170)

self.entry\_computer = Entry(self, textvariable=self.computer)

self.entry\_computer.place(x=380, y=170)

Button(self,text='修改',command=self.create\_user).place(x=320,y=220)

Label(self,textvariable=self.status\_name).place(x=305,y=250)

def id\_change(self):

if self.change\_student.get():

self.search\_user\_id = self.change\_student.get()

flag,stu = sql.search\_id(self.search\_user\_id)

if flag:

self.change\_info(stu)

self.status\_student.set('数据查询成功')

else:

self.status\_student.set(stu)

else:

self.status\_student.set('请输入需要查询的信息')

def name\_change(self):

if self.change\_student.get():

self.search\_user\_name = self.change\_student.get()

flag,stu=sql.search\_name(self.search\_user\_name)

if flag:

self.change\_info(stu)

self.status\_student.set('数据查询成功')

else:

self.status\_student.set(stu)

else:

self.status\_student.set('请输入需要查询的信息')

def change\_info(self,stu):

self.id.set(stu[0][0])

self.name.set(stu[0][1])

self.class1.set(stu[0][2])

self.math.set(stu[0][3])

self.english.set(stu[0][4])

self.computer.set(stu[0][5])

self.id\_change\_before.set(stu[0][0])

self.name\_change\_before.set(stu[0][1])

self.class1\_change\_before.set(stu[0][2])

self.math\_change\_before.set(stu[0][3])

self.english\_change\_before.set(stu[0][4])

self.computer\_change\_before.set(stu[0][5])

def create\_user(self):

if not self.id.get():

self.insert\_id = int(0)

self.status\_name.set('请输入修改的学号')

return

else:

self.insert\_id=int(self.id.get())

if not self.name.get():

self.insert\_name = 'NULL'

else:

self.insert\_name = self.name.get()

if not self.class1.get():

self.insert\_class1 = 'NULL'

else:

self.insert\_class1 = self.class1.get()

if not self.math.get():

self.insert\_math = int(0)

else:

self.insert\_math = int(self.math.get())

if not self.english.get():

self.insert\_english = int(0)

else:

self.insert\_english = int(self.english.get())

if not self.computer.get():

self.insert\_computer = int(0)

else:

self.insert\_computer= int(self.computer.get())

sql.delete\_id(self.id\_change\_before.get())

self.insert\_total = self.insert\_math + self.insert\_english + self.insert\_computer

stu = (self.insert\_id,self.insert\_name,self.insert\_class1,self.insert\_math,

self.insert\_english,self.insert\_computer,self.insert\_total)

sql.insert(stu)

self.status\_name.set('数据修改成功')

class HelpFrame(Frame):

def \_\_init\_\_(self,root):

super().\_\_init\_\_(root)

Label(self,text='关于录入界面').pack()

Label(self, text='可以录入所有信息为空的信息，但不建议，且学号具有唯一性').pack()

Label(self, text=' ').pack()

Label(self, text='关于查询界面').pack()

Label(self,text='默认为升序排列，可以根据学生的各类信息进行排列，并能通过快捷键以及鼠标右键实现一定的功能').pack()

Label(self, text='可以查看班级信息以及可以选择信息进行删除').pack()

Label(self, text=' ').pack()

Label(self, text='关于删除界面').pack()

Label(self, text='可以根据学号或者姓名对学生信息进行删除，学号是唯一的').pack()

Label(self, text=' ').pack()

Label(self, text='关于修改界面').pack()

Label(self,text='可以通过学号或者姓名来查询学学生信息，但查询名字只会出现第一位学生，按下修改键出现提示即成功').pack()

**四、Sql:**

from tkinter import \*  
import pymysql  
  
conn = pymysql.connect(  
 host='localhost',  
 port=3306,  
 user='root',  
 password='695273 ',  
 autocommit=True  
)  
  
sort\_stu = int(0)  
sort\_data = int(0)  
  
cursor = conn.cursor()  
  
cursor.execute("create database if not exists student1;")  
conn.select\_db("student1")  
  
sql1="create table if not exists students(id varchar(10) not null,name varchar(10),class1 varchar(10),math int,english int,computer int,total int,primary key (id))"  
cursor.execute(sql1)  
sql2="create table if not exists admin(name varchar(10),pwd varchar(10))"  
cursor.execute(sql2)  
  
def check\_login(uname,pwd):  
 cursor.execute("select \* from admin")  
 results = cursor.fetchall()  
 for na,pd in results:  
 if na == uname and pd == pwd:  
 return True,'登录成功'  
 else:  
 return False,'登录失败，用户名或密码错误'  
  
  
def add\_admin(uname,pwd):  
 cursor.execute("insert into admin values('{0}','{1}');".format(uname,pwd))  
  
  
def check\_username(uname):  
 cursor.execute("select count(\*) from admin where name = '{0}';".format(uname))  
 res = cursor.fetchall()  
 if res[0][0]:  
 return True  
 return False  
  
  
def all():  
 if sort\_stu==1:  
 if sort\_data == 0:  
 cursor.execute("select \* from students order by id;")  
 elif sort\_data == 1:  
 cursor.execute("select \* from students order by total;")  
 elif sort\_data == 2:  
 cursor.execute("select \* from students order by math;")  
 elif sort\_data == 3:  
 cursor.execute("select \* from students order by english;")  
 elif sort\_data == 4:  
 cursor.execute("select \* from students order by computer;")  
 else:  
 if sort\_data == 0:  
 cursor.execute("select \* from students order by id desc;")  
 elif sort\_data == 1:  
 cursor.execute("select \* from students order by total desc;")  
 elif sort\_data == 2:  
 cursor.execute("select \* from students order by math desc;")  
 elif sort\_data == 3:  
 cursor.execute("select \* from students order by english desc;")  
 elif sort\_data == 4:  
 cursor.execute("select \* from students order by computer desc;")  
 data = cursor.fetchall()  
 key = ('id','name','class','math','english','computer','total')  
 jsonList = []  
  
 for i in data:  
 jsonList.append(dict(zip(key,i)))  
 return jsonList  
  
  
def check\_id(id):  
 cursor.execute("select count(\*) from students where id = '{0}';".format(id))  
 res = cursor.fetchall()  
 if res[0][0]:  
 return False,"该学号已存在，请重新输入"  
 return True,'录入成功'  
  
  
def search\_class(class\_value):  
 cursor.execute("select \* from students where class1 = '{0}';".format(class\_value))  
 data = cursor.fetchall()  
 key = ('id','name','class1','math','english','computer','total')  
 jsonList = []  
  
 for i in data:  
 jsonList.append(dict(zip(key,i)))  
 return jsonList  
  
  
def insert(stu):  
 cursor.execute("insert into students values('{0}','{1}','{2}','{3}','{4}','{5}','{6}');".  
 format(stu[0],stu[1],stu[2],stu[3],stu[4],stu[5],stu[6]))  
  
  
def delete\_id(user\_id):  
 cursor.execute("select \* from students where id = '{0}';".format(user\_id))  
 res = cursor.fetchall()  
 if res[0][0]:  
 cursor.execute("delete from students where id = '{0}';".format(user\_id))  
 return True,'删除成功'  
 else:  
 return False,'学号为'+ str(user\_id) + '的学生不存在'  
  
  
def delete\_name(user\_name):  
 cursor.execute("select \* from students where name = '{0}';".format(user\_name))  
 res = cursor.fetchall()  
 if res[0][0]:  
 cursor.execute("delete from students where name = '{0}';".format(user\_name))  
 return True,'删除成功'  
 else:  
 return False,'姓名为'+ str(user\_name) + '的学生不存在'  
  
  
def search\_id(user\_id):  
 cursor.execute("select count(\*) from students where id = '{0}';".format(user\_id))  
 res = cursor.fetchall()  
 if res[0][0]:  
 cursor.execute("select \* from students where id = '{0}';".format(user\_id))  
 stu = cursor.fetchall()  
 return True,stu  
 else:  
 return False,'学号为' + str(user\_id) + '的学生不存在'  
  
  
def search\_name(user\_name):  
 cursor.execute("select count(\*) from students where name = '{0}';".format(user\_name))  
 res = cursor.fetchall()  
 if res[0][0]:  
 cursor.execute("select \* from students where name = '{0}';".format(user\_name))  
 stu = cursor.fetchall()  
 return True,stu  
 else:  
 return False,'姓名为' + str(user\_name) + '的学生不存在'  
  
  
if not check\_username("root"):  
 add\_admin('root','123456')